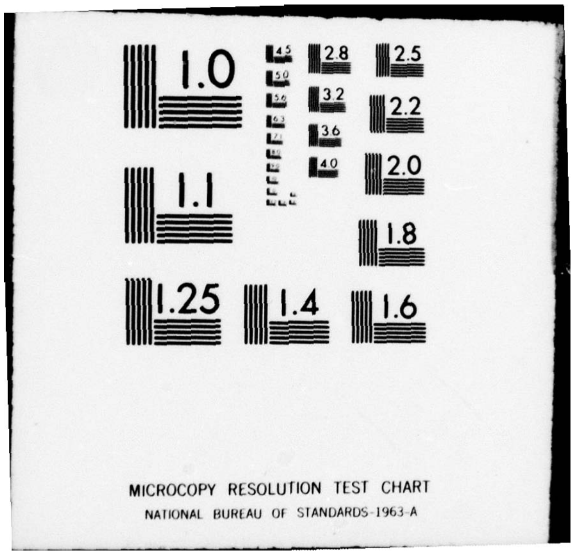
AD-A065 016 CALIFORNIA UNIV BERKELEY SCHOOL OF OPTOMETRY F/G 9/2
COMPUTER-CONTROLLER OPHTHALMIC REFRACTION. ANALYSIS OF A COMPUT--ETC(U)
SEP 77 E MARG, J BRASWELL, I LEE DADA17-72-C-2044
UNCLASSIFIED NL

| OF |
AD
A065016

END
DATE
FILMED
4 -79
DDC


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

COMPUTER-CONTROLLED OPHTHALMIC REFRACTION

Analysis of a Computer Network
vs.
a Central Computer

Annual Report

Jefferson Braswell, Imsong Lee, Elwin Marg

September 1977

Supported by

U.S. ARMY MEDICAL RESEARCH AND DEVELOPMENT COMMAND
Fort Detrick, Frederick, Maryland 21701

Contract No. DADA17-72-C-2044

University of California
Berkeley, California 94720

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (6)	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER (9)
4. TITLE (*and Subtitle*) Computer-Controller Ophthalmic Refraction. Analysis of a Computer Network ~~vs.~~ a Central Computer. versus		5. TYPE OF REPORT & PERIOD COVERED Annual Report. June 1976 - September 1977
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Elwin Marg, Jefferson Braswell, Imsong Lee		8. CONTRACT OR GRANT NUMBER(s) (15) DADA 17-72-C-2044
9. PERFORMING ORGANIZATION NAME AND ADDRESS School of Optometry, University of California, Berkeley, California 94720		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (17) 62778A (16) 3S762778A838 00 121
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Medical Research and Development, Fort Detrick, Frederick, Maryland 21701		12. REPORT DATE (11) 30 September 1977
		13. NUMBER OF PAGES 65
14. MONITORING AGENCY NAME & ADDRESS(*if different from Controlling Office*) (12) 65p.		15. SECURITY CLASS. (*of this report*) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (*of this Report*)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

computer-assisted eye examination, subjective refraction, standard optometric tests, case history, overlaying, optimum, system organization, multi-programming central computer, local processor, distributed system, central file system, communications network, secondary storage, floppy disks, DEC tape, software, hardware, configuration

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

This paper studies the projected organization and cost of a prototype computer-assisted eye clinic designed to supply one optometrist with an optimal flow of patients by freeing him/her from all tasks which do not require the professional skills of the optomotrist. Such a system is a logical extension of the current computer refraction project, which has demonstrated the feasibility of administering many routine optometric tests under computer control.

The organization and operation of the current system are presented as background for the design questions involved in the clinical prototype, and a user-

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

072 050

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

20. Abstract (continued)

oriented functional organization of the optimal system is then delineated. From these and other user requirements, a clear statement of the evaluation criteria and performance constrains for the optimal system is derived.

Major design approaches are considered, and first-order analysis singles out a distributed organization, which entails a network of small, inexpensive processing stations, is investigated in detail and the hardware/software configuration of the proposed system is discussed. Notable features of the proposed organization are the ease of maintenance via module replacement, the ease of custom-tailoring the system size by adding local processing stations assembled from standard hardware modules, the resilience of the system to local failures, and the incorporation of powerful but inexpensive LSI technology and highly economical floppy disk secondard storage.

Finally, a summary of a detailed cost study is included which appraised the cost of developing a prototype by itemizing the hardware and software development costs for each local station in the system. To this summary of direct costs are affixed the additional costs incurred in developing and operating the prototype.

Table of Contents

Forward

This analysis was made in 1974, before Refractor III had been developed. In light of this fact, the following remarks apply:

The Refractor III software system implementation differs greatly from the Refractor II system described briefly in Section 1.0.

The sharp drop in the cost of LSI logic in the past three years makes development cost estimates all the more decisive.

This fact, along with the product offering from Digital Equipment Corporation of the CLASSIC system (in extensive 32K PDP-8/A with dual floppy disk, console CRT, and hard copy unit) suggest that the distributed implementation of the clinic could be served by such stand-alone systems at each station communicating with the removable floppy disk medium as opposed to electronic interconnection to a central file.

ABSTRACT

This paper studies the projected organization and cost of a prototype computer assisted eye clinic designed to supply one optometrist with an optimal flow of patients by freeing him/her from all tasks which do not require the professional skills of the optometrist. Such a system is a logical extension of the current computer refraction project, which has demonstrated the feasibility of administering many routine optometric tests under computer control.

The organization and operation of the current system are presented as background for the design questions involved in the clinical prototype, and a user-oriented functional organization of the optimal system is then delineated. From these and other user requirements, a clear statement of the evaluation criteria and performance constraints for the optimal system is derived.

Major design approaches are considered, and first-order analysis singles out a distributed-function system architecture over a centralized scheme upon comparison of the design implications of each with the user-generated criteria. The distributed organization, which entails a network of small, inexpensive processing stations, is investigated in detail and the hardware/software configuration of the proposed system is discussed. Notable features of the proposed organization are the ease of maintenance via module replacement, the ease of custom-tailoring the system size by adding local

processing stations assembled from standard hardware modules, the resilience of the system to local failures, and the incorporation of powerful but inexpensive LSI technology and highly economical floppy disk secondary storage.

Finally, a summary of a detailed cost study is included which appraised the cost of developing a prototype by itemizing the hardware and software development costs for each local station in the system. To this summary of direct costs are affixed the additional costs incurred in developing and operating the prototype.

Generally, it is felt that this application of new technology offers considerable future savings and operating economies in the administration of optometric examinations and in other areas which will no doubt reap second generation benefits from the development of efficient distributed computing techniques and modularized systems.

1.0 CURRENT SYSTEM ORGANIZATION AND PERFORMANCE

The current computer-assisted eye examination project has been performing reliable and accurate subjective refractions and visual acuity tests in addition to maintaining and displaying patient files. The system has satisfactorily administered a wide range of refractor-based tests and soon will be capable of performing nearly all the standard optometric tests, the exceptions being those tests which require a high level of professional skill and human judgement. The computer also can generate a case history by asking the patient questions with a tape recorder and monitoring the answers with a response box, although this program has not been in frequent use. This demonstrated ability of a computerized system to correctly diagnose refractive error is a significant accomplishment in that it justifies any attempt to improve upon the overall performance of such a system whose underlying premise has been thus tested and proven.

Such improvements of the Refractor II system performance certainly can be made by refining test accuracy and increasing system reliability, but the most dramatic improvements are to be made by re-organizing the system for greater work-loads while maintaining the quality of the results at their current highly satisfactory level. It should be mentioned in passing that such a re-organization for improved patient flow should in no way impose undesirable inconveniences on patients or users. In the remainder of this section we will summarize the organization and mode of operation of the current system in order to suggest ways to improve system performance.

Figure 1 illustrates the simplified hardware/software relationships of the system which is in operation at the time of this writing. A system with the same basic organization, but with internal modifications and an improved refractor, will be installed in San Francisco in June, 1976. At present the system is designed to perform the following major functions:

Admit new patients;	(system creates new patient file and request preliminary patient data from operator, such as name, address, etc.)
Administer a case history;	(not currently performed)
Perform a subjective refraction;	
Print out case history;	(not currently performed)
Print out test results and final diagnosis;	
Add a case history or refraction to a past patient's file;	(performs the desired test and then inserts the results into patient's permanent file)
Perform various file operations;	(FIND a patient's file in the file directory, LIST the entire directory for the current data tape; DELETE a patient's file from the file directory, MOVE the temporary file onto the permanent file space allocated for the patient under test and INITIALIZE a new tape to be used for another

patient file data tape)

In addition to the above functions performed in the course of normal system operation, the system is equipped with various software development tools (Fortran compilers, absolute and linking loaders, assemblers, a hard disk, etc.) which are not involved in the execution of the dedicated system functions.

The variety of major system functions listed above is accommodated in the following manner: the "main program" is actually a central command decoder and related tape and input/output routines which are loaded into field 2 of main memory at system start-up and which reside there for the duration of system operation. These nucleus routines look to the user input keyboard for commands specifying which major function to activate and, upon recognition of a legal command, supervise bringing the software routines necessary for performing that major function from secondary storage (DEC Tape) into main memory as a block, where it occupies field zero or one.

Once the required set of routines has been loaded, the central program passes control to the start of the block and the major function is activated. This procedure is known as "overlaying" and is done because the total software needed to perform the entire set of possible system functions is much too large to fit in main memory at one time. When a given job is requested, only the code needed to run that particular job must be in core, and it is "overlaid" on the space where the previous job temporarily resides, fields zero or one. Examination of Figure 1 will reveal that the overlays on the system tape correspond with the primary jobs previously listed which the system is designed to accomplish. All routines for performing operations on the patient files data tape (not shown in Figure 1) are found in overlay 1. A patient files data tape must also be correctly mounted for the system to operate.

A typical examination of a new patient involves typing the command RUN. The central program will then ask for the patient's name and other preliminary data for the patient's file head, which is kept in the temporary file buffer along with the other results which accumulate in the temporary file in the course of an examination. The central program also sees to it that an entry is made in the file directory for the new patient and that permanent file space is reserved for the patient's file. When all tests that are scheduled to be run on the patient are successfully completed, the temporary file which contains the complete data for the patient will be written onto the patient's reserved permanent file space.

After the computer has set up the file and taken care of the preliminaries, it will ask if a case history is desired (yes or no) and if a subjective refraction is desired (yes or no). The central program then will bring in the necessary overlays to accomplish the specified tasks one at a time, and upon completion of all the jobs requested the temporary file will then be transferred to the patient's permanent file.

After this has been done, control will again be passed to the central program which will monitor the input keyboard for any further system commands. Typically, the operator will type "PRINT" and the overlays to print out a patient's file will be brought into main memory and executed, outputting the contents of the file. Another command has been recently added, "LDEV", which allows the operator to specify either the CRT display or the Teletype (for a hard copy) as the output device prior to typing "PRINT". When the output job has been completed the system returns to keyboard monitor status, and another patient can be processed.

In the event that a previously admitted patient is to be tested, the operator can type "ADD". This will serve to bypass the initial file creation and preliminary data acquisition necessary for new patients, and will instead simply ask for the patient's name and whether or not a case history or subjective refraction is to be performed and the results added to the patient's file.

The following summary of the major features of system organization is based on the above brief description of the current system operation.

1. Only one major functional job can be active at a time and it must run to conclusion before another major task can be initiated.

2. Only one patient can be processed at a time. This follows from the previous point, and also from the fact that all test routines report to a single temporary file buffer.

3. The previous two constraints are both software-oriented. With a re-organization of the software only, it would seem possible to overlap the file creation of one patient with the subjective refraction of another, the case history of a third, and the print out of a fourth.

In view of the last point it should be stated that such a mode of operation is not, in fact, totally hardware-independent. All the external devices certainly are capable of simultaneous operation, but the speed of the processor places an upper limit on the real-time multiprogramming power of the system. This problem is greatly reduced in the present hardware configuration since the random logic interfaces perform most of the real-time control of the devices upon simple command from the device handler programs in the central computer. Nevertheless, such a software re-organization should not be undertaken until an adequate measurement of the current degree of processor utilization is made and extrapolated to account for demands on the processor in the re-organized system.

A further constraint would be caused by overlay swapping in and out of main memory, an essential feature of a multi-programmed configuration using the current amount of main memory (12K). This will cause time delays much larger than speeds of program execution account for since the secondary storage device involved in the transfer is a Dec tape drive, a unit with long access times.

In short, improvements in system performance can be made by doing more jobs in parallel, yet such a step is not absolutely straight-forward. An analysis of the type of patient flow which would be optimal to an optometrist using such a system must be made, and based upon that, a consideration of the computer system most appropriate to the task can be undertaken.

2.0 CHARACTERISTICS OF AN OPTIMAL COMPUTER-ASSISTED EYE CLINIC

In the previous section we described the operation of the Refractor II's experimental system which has shown the feasibility of greatly reducing the human work-load involved in the processing of patients in an eye clinic by performing most of the repetitive procedures with the assistance or under the control of a digital computer system. In considering improvements in the Refractor II system it would be wise to project a target system which would furnish an optometrist using such a system an optimal flow of patients whom the optometrist would examine and finally diagnose on the basis of the automated tests and his/her own observations. The objective of this target system would be to most effectively utilize the professional skills of the optometrist in terms of the number of patients he/she could examine in a period of time (e.g. in the course of a day). Should such an arrangement become desirable or necessary, as the projected shortages of optometrists suggests, improvements made on the Refractor II system with this target system in mind would be a prudent investment of research and development energies.

The most efficient use of an optometrist's time is made if:

a) the optometrist performs only those aspects of an examination which can be done only by an optometrist;

b) the rest of the clinic is supplying the optometrist with patients to examine at an optimum rate.

In order to get a precise assessment of the times involved in the division of labor described here, a statistical study was made of 15 optometrists engaged in private practice. The time spent on various phases of the entire procedure of examining a patient was recorded. This consists of such things as admitting a patient, obtaining a case history, testing for pathologies, testing visual acuities, performing an objective and a subjective refraction, consulting with the patient about his symptoms, recording the diagnosis, specifying the prescription, choosing frames, and discharging the patient. Based upon this empirical study (which is available separately) and the performance of the existing computer system, an optimal computer-assisted eye clinic has been projected. In this clinic, the organization of the target system frees the doctor from most of the mundane, time-consuming aspects of the examination and enables him to quickly but accurately diagnose each patient. As projected, the number of patients processed by up to six optometrists in the course of a day could be seen by one optometrist with human and computer assistance.

Figure 1 depicts a possible floor plan for such a clinic, whose mode of operation will now be described from the user point of view. Upon entering the clinic, patients see a receptionist to be admitted. The receptionist operates a terminal which creates and manages patient files for him and accepts preliminary data; he also has access to the secondary storage medium employed so that he can insert back patient files, not all of which will be on-line. The patient is issued a magnetic card which identifies him to all the local stations, and directed to the case history booths. The patient is actually free to go to the automated test stations (case history, visual acuity, objective VEP refraction, subjective refraction) and the pathology station (technician equipped with terminal for reporting results) in whatever order is most convenient as long as all these stations have seen the patient before he/she sees the doctor. The doctor is also equipped with a terminal for printing the case history and test results and for entering his diagnosis and prescription after examining the patient. The patient then goes to the optician to choose frames and be fitted, and finally to the business office.

This floor plan represents the configuration believed to optimize the patient flow through the system and most effectively utilize the doctor's professional skills, and the replication of various test stations in Figure 1 reflects the examination times at each station and the need to balance the flow of patients through the system with the doctor's examination time. A more detailed user-oriented functional organization than described above has been planned out which analyzes the user commands, system functions, and data transfers projected for each station; this structural diagram is available on request.

As a final step in describing the characteristics of an optimal computer-assisted eye clinic, Table 1 states the evaluation criteria and performance constraints which apply to any specific computer system implementing the projected system functions pursuant to additional user-oriented performance requirements.

3.0 DESIGN ALTERNATIVES AND THEIR IMPLICATIONS

In designing a system to handle the scale of operations implied by the clinic just described, a number of approaches are possible. The chief organizational differences in the various alternatives lie primarily in the degree of centralized and distributed functions that these alternatives possess. Table 2 presents a rough sketch of the spectrum of conceivable system organizations ranging from totally centralized to totally distributed. The purpose of this section is to arrive at the most appropriate overall system organization suitable for expansion of the present system up to and including the optimal clinic. A detailed study is then made of the possible implementations of this overall organization in the next section.

The design approach which immediately comes to mind as a possibility is simply to take the current organization and expand it so that it can handle the heavier service requests and data transfers to be expected in the enlarged system. Instead of one user terminal for entering commands and receiving information from the computer, there would be six: one receptionist, two technicians (pathology), one doctor, one optician, and one business office. There is also the possibility of needing a separate terminal for software development which could function concurrently with normal system operation. Likewise, instead of controlling one set of tape recorders, slide projector, and refractor, the system would control up to four refractors, 10 tape cartridges, six tape cassettes, five slide projectors, and eight answer boxes. With one receptionist, three case history stations, one visual acuity station, two objective refraction stations, two subjective refraction stations, two pathology detection stations, one doctor, one optician, and one business office, there are 15 different jobs that need to be serviced concurrently. It also should be mentioned that all these jobs are tied very strongly with I/O and file operations and are only nominally involved in internal computation.

It certainly is possible for a digital computer to accomplish all the above tasks with reasonable response time if the computer is powerful enough. Even with re-entrant code (code which can be used simultaneously by two or more running jobs with separate data fields) it would not be economical to keep all the necessary software in core, and a high speed secondary storage capability, such as hard disks or drums would be necessary for swapping jobs in and out of main memory. Double-buffering (operating on one part of a buffer while another part is involved in a data transfer) and a Direct Memory Access channel also would be essential so that jobs could be loaded while the Central Processor was executing other jobs. In addition to these requirements, a sizeable operating system is required for scheduling tasks in this time-sharing/multiprogramming environment and for permitting easy re-configuration of the system by the user.

More software overhead is encountered in the need for a file system with adequate levels of file protection. Finally, the processor power (appraised in terms of the cycle time and the instruction set) would have to be considerable to handle the above work load, and the I/O architecture of the system must permit the processor to easily address and control a large number of external devices. A conventional mini would be hard-pressed to meet these requirements taken together and a medium-scale computer, at least, would be needed here. Figure 2 depicts the general organization we have just described, with the powerful computer at its center.

Instead of a single, large computer performing all the system functions, various jobs could be broken into classes, with a less powerful processor assigned to each class of job. For example, a processor could manage files and secondary storage, a second could execute programs, and a third could handle all I/O. The complexity of the processor executing programs is still considerable, however, and this manner of distributing system functions introduces additional intricacies in communications between processors which are strongly dependent on each other and are in constant interaction.

A more natural grouping of system jobs might arise from assigning a processor to manage a restricted class of test functions associated with each test station. By virtue of this arrangement, demands on any one processor are minimal compared to the one-computer-all-jobs configuration, and it no doubt would be possible to use microprocessors as the basic processor unit. The extensive data transfers necessary for swapping jobs in a time-sharing environment or merely for overlaying in a large software/small memory situation would not be necessary in a configuration with a small processor dedicated to each test station (or "job"), since the software related to each specific station is small enough to be manageable in a limited amount of memory for each processor. Such a scheme for distributing the functions of the system to independent processor stations would, furthermore, very conveniently solve the multi-programming problem, since each job which would previously have vied with 10 to 15 others for the attention of a central computer now has its own processor to execute its own code. One new implication of this mode of organization is that the typical station has no secondary storage under its immediate control (in a few cases this is not true), and all patient file operations must be transacted with a central file system which all stations report to in some fashion. Thus, although the software overhead at local stations would be at a minimum, a fairly intelligent network handler and file controller must be implemented at a central file station in addition to the simpler file handlers at each local station which relate to the network handler. However, the data rates involved in this communication network are low in comparison to the rates needed to sustain program transfers as well as data transfers in the centralized, multi-programmed alternative.

On the hardware side of the picture, it should be fairly obvious that the local processing stations are actually reductions and simplifications of the system currently operating, eliminating secondary storage, unrelated software, and unnecessary hardware interfaces where applicable. As for the network linking the local stations with the central file, two possibilities stand out: a bus which addresses all stations simultaneously but with only one pair actively communicating at any moment, and secondly a ring structure in which messages are passed consecutively around the ring until they reach their destination and are removed from circulation. The overall organization of this alternative is presented in Figure 3 which shows the local processing stations for the automatic test stations and the user interactive stations tied together in a network which contains the central file as one of its modes.

These two alternatives, as described above, both meet the overall system performance requirements in their capabilities for digital control and information processing. The normal user interface and system development user interface requirements both are primarily matters of having the software to support these requirements, and the general software differences in the two alternatives have been mentioned. The software responsible for the implementation of the user-oriented high level command language is judged to be roughly compatible in terms of development and memory costs, although it is worth noting that the hardware in the centralized alternative must be capable of servicing user requests from seven terminals and eight test stations in less than two seconds.

However, with regard to the other criteria and constraints, there is a marked difference in the two configurations. If faults develop in local stations which generate bad input to the centralized computer, the probability that errors would be propagated through the system and even cause it to crash can not be disregarded. Naturally, any minor fault in the central computer would have very undesirable effects on the entire system. In the distributed configuration, a fault in one of the stations would have virtually no effect on the rest of the system. If faults developed in the central file system, the local stations would be empowered to temporarily buffer file messages until the back-up file system was operational. Redundancy in a central file processor is a much simpler matter than providing a back-up capability for a large, general-purpose computer.

When faults do bring the large computer in the centralized alternative down, the whole system is down until the fault is located and corrected. Maintenance in the distributed system is more convenient; the hardware module compatibility among test stations permits replacing a standard module with a back-up module while the defective module is being repaired, regardless of the local station involved. Furthermore, a local processor could fairly easily add some degree of error diagnosis and reporting to its limited duties, whereas requiring a central computer to monitor errors in the entire system in addition to its normal work load might add noticeably to the performance required of such a computer.

Another serious shortcoming of the centralized scheme is the fact that unless a tremendously powerful computer were used at the outset, expanding the capacity of the system from one stage of development to another would at some point require getting a completely new machine to replace the computer which was initially installed and whose capacity had been saturated. To begin expanding the current system's powers by acquiring a computer capable of handling the processing load envisioned at the final stage of the system's development would be the only way to avoid this difficulty, but this is risky for a number of reasons:

1. The course of development would be too rigidly determined. Should intervening events suggest or dictate a different final system than envisioned, the initial system may be mismatched with ultimate objectives.

2. Planning that far into the future with conventional technology locked into the system would prevent future

stages of development from benefitting from future developments of the electronics industry.

3. Such an initial step would require a considerable capital investment.

The distributed configuration, on the other hand, is ideally suited to flexible development plans and the incorporation of cost-reducing technological innovations currently available or expected in the near future. Should it become desirable to add another test station in order to balance patient flow through the system or to increase the system's capacity, another processor can very easily be added to the network. The hardware changes in the rest of the system needed to effect the addition of a local station are minimal, differing slightly with the type of network. If the network is the "star" type (each local processor connected directly to the central node), then the interface at the central arbitrator simply needs to have enough slots for at least 20 interface cards buffering the lines to the local stations. Depending on the length of the cable involved, some form of line conditioning or repeaters might be necessary. If the network is a "ring" (each local processor connected to two adjacent stations, one to which it passes or relays messages and the other from which it receives messages intended for it or another station around the ring), all that is involved is making a momentary break in the ring and inserting a station between two previously adjacent ones. Because of the relative proximity of adjacent stations, no special transmission measures are expected in the case of a ring network.

The hardware addition of a local station of any kind also is simplified by the fact that the hardware modules which comprise a local processor station can be standardized for the entire system, ruling out the need for any design work if such an addition is contemplated. All the user need do is call on the supply of extra modules kept on hand for the system maintenance and expansion. These modules will be enumerated in the next section.

An aspect of the future-oriented flexibility of the distributed system equal in importance to the ease of reconfiguration the hardware for specific work-loads is the incorporation of advanced technological innovations in its design and the preservation of future options to make use of projected technological capabilities. Reference is made primarily to the success of large-scale integration (LSI) in drastically reducing the chip-count and design fabrication costs of digital systems while clearly increasing the range of applications of systems using programmable LSI logic in place of inflexible random logic. The architecture of the distributed system is such that the performance required of the processor stations can readily be met by a computer which is based on a microprocessor. Furthermore, it is anticipated that most of the device interface modules could eventually be replaced by LSI programmable logic should that step become desirable for any number of reasons, chief of which might be resultant efficiencies if manufacture of the units were a consideration. In any event, the hardware configuration of the distributed system permits the substitution of functionally equivalent connector-compatible hardware modules which have been internally redesigned to incorporate significant advances in state-of-the-art LSI technology.

On the basis of the preceding analysis alone, it is apparent that the distributed alternative is much more suited to this particular application than is the centralized version. The network of smaller processor stations possesses a number of advantages over the central-computer concept, and, more decisively, the centralized alternative fails to meet several key evaluation criteria - notably, the stipulation concerning system reliability (Section 4.0). The centralized alternative also is judged deficient in light of the system development criterion for ease in reconfiguring the system (Section 3.0) and, more importantly, in light of the future option criterion specifying the ability to add stations to improve the performance of the system (Section 6.0). Finally, the centralized alternative is judged poor with respect to the degree of hardware modularity and debugging simplicity it affords, a maintenance criterion (Section 5.0). Table 2 sums up the comparison of the two major alternatives in terms of their primary system features.

In short, although a more extensive analysis obviously is possible, the above considerations are sufficient to rule out a centralized computer architecture as the basis for expansion of the current system up to and including the projected optimal one-optometrist computer-facilitated eye clinic. Even though this decision was made largely on a performance basis, the distributed alternative is judged thoroughly cost-competitive with the centralized version, given the orders-of-magnitude difference in the cost of large-scale general purpose computers and the cost of microprocessors.

With the scope of the feasible systems for this application limited to some form of distributed network of small processors, we now can look at the necessary structure of the hardware/software relationships and the possible implementations of such a system in more detail.

4.0 PROPOSED SYSTEM ORGANIZATION

At the heart of the proposed system is a multi-access file sub-system which interacts with the local processing stations via the communications network. The local processing stations, the communications network, and the central file system can be in various modes or states in the course of system operation. Perhaps surprisingly, it is not necessary for the three major sub-systems mentioned above to be simultaneously active for the system to operate, although this is not usually the case. The system works perfectly well if only one local station is active; to the central file system the status of stations resembles the status of remote terminals in a time-sharing network. Similarly, the local stations, once loaded, have the ability to operate without the central file system or the communications network (messages to the central file are buffered for later transmission if either of the two sub-systems are down); only in the event that information definitely is needed from the central file would the normal operation of a local station be affected by inactive file or network sub-systems.

Generally speaking, the entire system could be in one of several major states depending upon the states of the system's components:

1. Power-off;

2. Central bootstrap: (Central file system station power-on and initialization of central message processor)

3. Central ready:
   (Central file system station ready, local stations power off)

4. Load ready:
   (Central file system station ready, local stations power-on)

5. Load Execute:
   (Loading of local stations in progress)

6. System ready:
   (All stations loaded and ready)

7. System operate:
   (Initiated by the first station to request a central file operation when the state is System Ready; this state covers the normal operation of the system - i.e., execution of software at local processors and transfer of information between local processors and file system)

8. System error:
   (Entered when local stations report error condition or when error is detected by central monitor; system will try to recover (soft error) and log condition in file and on printer. If error is not recoverable and is

fatal, system will attempt to enter one of two diagnostic/alert states: Local Fail or Central Fail)

9. Local fail:
(Local station has gone down, operator is alerted, and rest of system continues normal operation)

10. Central fail:
(Some aspect of central system is down; operator is alerted by alarm; local stations buffer or wait; back-up facility exercised while problem corrected)

11. Development interrupt:
(System development station wishes to halt execution at a local station for software testing or development at that station; rest of system continues at normal operation while software at the interrupted station is dumped, edited, loaded, etc.)

12. System close-out
(Entered at end of the day when operator signals completion of normal operation; cumulative statistics updated and summaries of system status logged and printed out; patient file directories modified to reflect current files; hardware exercised for routine fault detection)

In the remainder of this section a closer examination of the three primary sub-systems mentioned above will be given: the local processor station, the file system, and the communication network. The communications network can be thought of as the intersection of the local station sub-systems and the central file sub-system; it consists of the network interface modules found at every station and any extra devices needed to regulate the multiple attempts to access the central file manager with this network. The local processor station is a standard modular, bus-oriented small computer with standard and custom interface boards, processor board, and memory boards. The central file system is the most sophisticated of the sub-systems, containing the apparatus necessary to service and schedule multiple messages to and from all other stations.

4.1 Local Processor Hardware Configuration

Every local processor, regardless of type, possesses the following hardware modules:

a. Processor:
(either a board supporting the Intel 8080 or the recently announced PDP-8/A processor board module)

b. Random-access semi-conductor memory:
(either organized into 8-bit words for the 8080 or the 12-bit by 1, 2, or 4K memory boards supplied by DEC for the PDP-8/A)

c. Direct-memory access (DMA) channel:
(to enable file transfers while the processor performs other tasks)

d. File system network interface:

(enables all communication between local station and central file station via network)

e. Device interface(s):

(used by local processor to control or monitor external devices deployed at that station)

The types of devices and device interfaces used at the local stations varies from station to station, depending upon the functions to be carried out at each station. The following list enumerates the types of device interfaces which various stations will employ:

e.1 Magnetic card reader interface:

(custom design; all stations except central file system and system development station need a magnetic card reader for identifying patients as they progress through the system)

e.2 Tape cartridge interface:

(custom design; tape cartridges are used to issue relatively short and few messages to patients at these stations: visual acuity, objective refraction, subjective refraction)

e.3 Tape cassette interface:

(custom design; digital cassettes are used for issuing a large number of possible messages to patients at the case history station) e.4 Answer box interface:

(custom design; answer box is used at all automatic test stations to record patient responses to taped questions: case history, visual acuity, objective refraction, subjective refraction)

e.5 Slide projector interface:

(custom design; these stations control a slide projector to perform tests: visual acuity, objective refraction, subjective refraction)

e.6 Refractor interfaces:

(custom design; the refractor interfaces are a large number of smaller boards which control the stepper and axis motors in the refractor; these interfaces are located in a cabinet separate from the local processor cabinet, which contains a single board that communicates with the refractor interfaces on a bus of their own in the remote cabinet; needed at objective and subjective refractor stations)

e.7 Display interface:

(custom design; light-emitting diode (LED) display is used at the refraction stations to reveal the particular test and lens combination administered to the patient)

e.8 Keyboard/printer interface:

(custom design - 8080; OEM module - PDP-8/A; this device is the primary device involved at the user-interactive class of stations; receptionist, pathology detection, doctor, optician, business office, system development)

The only remaining digital hardware which enters into the configuration of some of the local processor stations is a floppy disk and floppy disk controller. This secondary storage capability at a local station is the exception rather than the rule, and is introduced only because it greatly simplifies the operation of that station, renders a station functionally independent of the central file, or eliminates an over-bearing demand on the central file system by a local station.

The stations which need local secondary storage because of the preceding considerations are the receptionist, business office, and system development facility. The receptionist needs back patient files physically available for retrieving previous patient records when patients return to the clinic. Keeping all back files on-line is possible but very cost-prohibitive. The compromise is to keep a dictionary of all past patients on some secondary storage medium which the receptionist can search in order to identify the patient's past day file; these files are off-line but on a readily insertable medium. If the search is unsuccessful, the system will request a patient directory which covers an earlier period of time (in units of six months or a year) than the span of time covered by the directory just searched. If the search was successful, the system will inform the receptionist which past day file to insert in order to transfer the patient's records to the current day file, which is maintained in the central file. This procedure is necessary because it is the receptionist station which creates an entry in the current day file for incoming patients; when other stations report to the central file concerning a patient they assume the patient's file has been inserted into the current day file which the central file manages.

It is expected that local secondary storage at the business office will initially be helpful and soon prove to be essential for the accommplishment of data processing tasks. These duties include payroll preparation, inventory control, maintaining patient account files, maintaining corporate accounts (for insurance reimbursement) and accounts payable files, and billing and mailing list software. Depending upon the storage capacity of the on-line central storage medium, this rather extensive group of programs and data conceivably could be kept on-line in the central file (no manual insertion is possible during the System Operate state, or normal operation of the system). This presents a growing problem, however, since over the course of the clinic's operation the data files mentioned above will in most cases grow without bounds and will exceed the capacity of an economically modest on-line central file facility.

This limit is real and readily approachable if the storage medium is a hard disk, and certain to be reached almost immediately if the storage medium is the more economical floppy disk.

For this reason, local secondary storage for the business office is included in the hardware configuration.

Manual insertion of segmented files independent of the central file thereby is possible just as in the receptionist station. The business office station can obtain the expenses incurred by the patient when they happen (via the network), update the appropriate files locally, and whenever the patient decides to visit the business office his account will be current. This arrangement has the additional benefit of not needing the central file or the network for the independent execution of billing, payroll, mailing and other software; also it substantially reduces the load on the central file system for the same reason.

Finally, the system development station needs local secondary storage for the same general reasons: large amounts of software are involved (compilers, loaders, special software for re-ordering test routines or station configuration, etc), and execution of this type of software in a small computer environment generally involves exchanging sections of code between main memory and secondary storage. Furthermore, the various versions of the software routines used in the local stations will need to be on-line or readily insertable for editing or building the software for these stations. Some form of inexpensive, versatile secondary storage medium therefore would be needed here as well.

The storage medium for the three local stations which need this facility could be chosen from a number of specific options; digital cassettes, floppy disks, low-cost magnetic tape (such as DEC tape), or hard disks. For the number of such storage facilities contemplated, and the order-of-magnitude of the storage requirements involved, we can rule out hard disks because they are too expensive (especially in these numbers) and offer more capacity than is really necessary for these applications. IBM performed a rather extensive study of floppy disks and digital cassettes and concluded that digital cassettes fell down in the comparison because the reliability of floppy disks was judged to be superior to that of digital cassettes. This leaves us with a choice between floppy disk and DEC tape.

The storage capacity of floppy disks and DEC tape is compatible. One floppy disk has the gross storage capacity of approximately 2.6 million bits. There is roughly a 3% error-checking overhead in a standard sector/checksum error-checking format, leaving in the neighborhood of 2.5 million bits for actual information storage. DEC tape, on the other hand, has roughly 850,000 lines at 10 tracks per line; extensive redundancy of data and timing tracks leaves only three effective information tracks per line, however.

Thus, the total useable information capacity of one reel of DEC tape is 3x850,000 = 2.55 million bits, or virtually the same as one floppy disk.

Comparison of the real-time performance characteristics of the two mediums reveals some differences, however. The data transfer rate of a typical floppy disk (the Shugart SA901 Disk Drive) is 248,000 bits/second; the data rate off the DEC tape is nearly 100,000 bits/second.

Access times of the two devices diverge sharply. The typical floppy disk rotates at 360 rpm, or one rotation every 166.7 milliseconds; this implies an average latency once a desired track has been reached of 83 milliseconds. Track-to-track access timeis 10 milliseconds, and an additional 10 milliseconds is required for settling. This gives a total average access time of roughly 100 milliseconds. DEC tape access involves a linear search and is much more time consuming. The total length of DEC tape is 250 feet, and the tape transport moves the tape at speeds of 93 plus or minus 12 inches per second. To search the entire length of the tape would therefore take 32 plus or minus 5 seconds. Average access time depends primarily upon the statistical distribution of the blocks of data on the tape; if they are inclose proximity, the data transfer can be effected in two or three seconds on the average; if the files are more randomly distributed, or if consulting the file directory for the location of a file is necessary, much longer access times are involved.

As a last consideration in comparing the performance of these two devices, the reliability of the DEC tape medium for soft error recovery is greater than that of the floppy disk because of the DEC tape's extensive track redundancy. However, the floppy disk medium's soft error recovery capability is certainly adequate and efficient: repeated passes of a track are made if sector check-sums indicate a bit has been dropped or added. If the discrepancy is not rectified after a certain number of tries, the floppy disk has a hard error. Manufacturer specifications assure a minimum of 10(6) contact passes before hard errors might be expected in the floppy disk itself.

When the response times of the two devices are considered along with their costs, the floppy disk is by far the most desirable system. A dual-drive floppy disk can be obtained for around $1200, and the controller can be designed and built for around $600; a Dual DEC tape, on the other hand, goes for $5000 and the necessary controller (which can control up to four dual drives) costs $4500. For these reasons, the secondary storage medium included in the design of the local processor stations - and the central file system as well - is a dual floppy disk drive.

The digital hardware at each local station is a collection of modules which forms an operational unit appropriate to the type of task a station is intended to perform, and yet the basic hardware organization of each station is highly similar. A relatively self-sufficient modular computer runs programs initially supplied to it by the central file system at system start-up; these programs control external devices through standardized device interfaces and communicate with the central file through a standardized file system interface. All these hardware modules are interchangeable throughout the system - it simply is the particular collection of modules at a station which distinguishes that station's overall function from that of another station.

4.2 Local Processor Software Organization

Software at the local processing units is organized into the following categories:

1. Local monitor:
   (Responsible for handling interrupts and elementary scheduling of tasks)

2. Station-dependent software:
   (Appreciable block of code which embodies the various functions which a particular station is designed to accomplish)

3. Device handlers:
   (Standardized software routines correlated with specific hardware device interfaces which the routines control; the device handlers serve as the effectors of

device .perations requested by station-dependent software or other higher level code)

4. File handler:
(Standardized software package which the local software uses to relate to the network interface module and ultimately to the central file system; the file handler coordinates the transmission or buffering of messages initiated by the local station and the reception of messages from the central file)

5. Error reporting routines:
(Logs error notifications with the central file)

6. Buffer:
(Part of local memory is allocated to buffering messages; the file system interface normally will deposit and pick up messages in this area via the Direct Memory Access channel or with the assistance of the file handler)

The station-dependent software generally falls into one of two main classes: programs for administering the automatic tests and reporting the results, and programs for interpreting user commands from a keyboard and entering or obtaining information from the central file. These two classes are the natural consequence of the fact that nearly all the stations are either automatic test stations or user-interactive stations. There are, in addition to software of those two general classes, several special software packages needed at stations with unique functions, such as the business office and the system development station. Another important block of software which is somewhat unique is the code which oversees the preparation and loading of local station software, the state transitions of the system (e.g. Load Execute to System Ready, System Operate to System Error), and communication with the supervisory system operator.

Figure 4 presents the generalized format of a typical local processor station to reveal the main hardware/software relationships at a glance.

4.3 File System Organization

The various file management functions contained in the current system are distributed among all the stations in the proposed system. These file operations cover a number of data transfers which range in size from a few bits to entire files or programs. During system loading at start-up the central file system will be engaged in extensive program transfers to all stations, most likely one at a time. During normal operation, however, the data transfers between local stations and the central file will depend on the kind of function each station performs.

When a local station does call upon the central file in the course of normal operation to either retrieve or deposit data concerning a patient, at no time is the local processor in control of the secondary storage medium which holds the on-line day file the processor wants to access. Rather, one section of the station-dependent software formulates a message in a special file system language internal to the system which identifies the patient, requesting station, the symbolic name of the information to be transferred, the direction in which the transfer is to be made, and the information itself if the transfer is from local station to central file (write). This message is built in the buffer area of the local station's memory; when it is complete the file handler software routine is informed of its whereabouts and is asked to see to it that the message gets to the central file. The file handler does this by calling upon the hardware in the file system interface, which waits until the communication network is free and then dumps the message out onto the network using the DMA channel. The file handler checks back with the file system interface until it sees that the transfer is underway, whereupon it returns control to the local monitor. If a certain time limit is exceeded and the transfer has not been initiated, the file handler assumes the network is inactive and keeps the message on its list of work to be completed when the network is responding. Finally, the file handler does not scratch this message from its list of unfinished business even if the transfer is initiated; it waits until it receives confirmation of reception from the central file message processor.

This mode of communication between local processor and central file means that the local processor must have the intelligence to formulate such a message; it also means, however, that the central file system station has to have the intelligence to interpret the message and then go about the actual business of accessing the floppy disk and correctly performing the data transfer. A number of local stations might well request central file access at almost the same time, and so the central file station will need to buffer messages and schedule the processing of multiple file tasks. Furthermore, the message processing facility can readily implement the necessary file protection measures and priorities for the local stations which want service.

Analysis of the projected peak loads which the file system might be required to handle indicates that the technology intended for the system is quite capable of performing the file operations with reasonable response times. In the optimal system, up to 16 different stations could conceivably request central file service simultaneously. If the priority scheme were a simple round-robin procedure, the message processor would examine a message in the queue, identify the requesting station and the type of file operation desired and, provided the station was allowed that type of access, carry out the transfer. (Requesting stations refer to file items by a symbolic address based on patient ID and logical name of the record, segment, or entry to be transferred; the central file message processor arrives at the exact floppy disk addresses of the transfer upon consultation with the patient directory maintained in memory as well as on the disk. The fixed format of patient records allows quick computation of real disk addresses using a base address/displacement method). After a transfer has been initiated by the message processor (send message tagged at the head of outgoing data and floppy disk controller given the start and end addresses), the message processor can prepare the next data transfer requested by another station in the queue while the previous transfer is underway. The time needed to process a message in the above manner (no priority considerations) would be roughly equivalent to the time involved in accessing the floppy disk and accomplishing the transfer.

The through-put of the floppy disk is well above the worst case load: if all 16 stations are in the queue, up to 10,000 bits of data might ultimately be transferred. The processing of this data is broken up by the accessing of different continuous blocks on the disk for different requesting stations. Recalling that each access takes 100 milliseconds on the average, this implies a total time required to process a peak load with overlapped message interpretation and file transfer of 16x100 msec + 80,000 bits/240,000 bits per sec (disk data rate) = approximately 2 seconds.

4.4 Communications Network

A number of options are possible for the network which links the local stations with the central file. Because the local processor stations will be some distance from the central file station, transmitting data in parallel on more than one line will involve some critical timing measures to avoid clock skew, and for this reason bit-serial transmission is favored.

If the "ring" network is employed, additional time must be added to the central file message processing and data transfer through-put times previously listed in order to arrive at the effective response time of the network and central file from the local station's point of view. This is because messages in the ring network encounter delays in travelling from station to station before reaching their destination. At a 1 Megabit serial transmission rate, network delays amount to a more critical constraint on the effective data rate than does the through-put of the central file station.

Two solutions present themselves: first, if a local station's request for central file service is to be accomplished in its entirety before another station is serviced, and if large data blocks are involved, then a star network with bit-serial transmission to each station from the central station would be effective. If, on the other hand, large messages such as patient file transfers between the doctor's station and the central file were broken up into smaller messages and interleaved with other messages around the ring, a station could receive data at rates of over 30 characters/sec (terminal print speed) and be insured of rapid initial response from the central file. It should be pointed out that the data from the central file are not routed directly to the terminal; the local processor accepts the data and formats it for terminal output.

Although it is well within the current technology to achieve the network and file system performance goals, these sub-systems will require extensive hardware and software design efforts to integrate these sub-systems into the smooth functioning of the entire system. The actual choice of message format and syntax as well as file system message scheduling policies must wait until detailed analysis of these features can be made in light of specific system implementations.

5.0 GENERAL RECOMMENDATIONS AND SUMMARY OF DETAILED COST ESTIMATES

Two hardware configurations were studied as possible implementations of the projected system. One involved the PDP-8/A, a recently announced modular processor board from Digital Equipment Corporation in Maynard, Massachusetts. The other version utilizes the Intel 8080 microprocessor as the basic processor element. The cost of the PDP-8/A is listed as $572 in quantities of 100, which puts it in the neighborhood of $1000 in single quantities. The 8080 is around $300 and is expected to drop significantly. The 8080, of course, would require additional chips to latch output addresses, multiplex data lines, and drive busses. The extra parts and the wire-wrap costs bring its estimated cost to $800 for each unit after an initial $400 for design.

If the PDP-8/A were selected, the Direct Memory Access module, the Memory modules, and the Keyboard/Printer Interface module would also be obtained from DEC; choice of the 8080 entails design and construction of these modules. The majority of hardware components are common to both systems, regardless of processor choice (i.e., custom designs), and this and the fact that the cost of the DEC modules is on a par with the cost of building the modules independently brings the hardware costs of the two versions into almost identical figures.

The processing power of the two versions is judged to be roughly equivalent for this application; the PDP-8/A is slightly faster than the 8080, but the instruction set of the 8080 is more extensive than that of the 8/A.

It is reasonable to expect the 8080-configured processor to be more reliable in the long term since the 8080 processor involves a substantially reduced chip count in comparison to the DEC module. Other maintenance costs are judged to be equivalent in view of the fact that DEC does not anticipate a comprehensive service contract arrangement for the modules, which they see as primarily geared for OEM markets and not for the end user. Maintenance costs for the rest of the system render minor differences in the processor reliabilities insignificant.

Because the PDP-8/A is compatible with all other PDP-8 software, the most notable difference in the two implementations lies in the degree of software development which each version requires. The existing 8/E software would no doubt require alteration, some of it extensive, but the man-hours needed to adapt the existing software to the projected system number less than the effort involved in writing the code for the 8080. This seems to be the primary advantage of the PDP-8/A. Both systems offer high level languages (PLM for the 8080), assemblers, and compilers - although here again DEC seems to offer more extensive software support than Intel.

The detailed analysis of the system proceeded in this manner: the function of every station is delineated, and the software routines and hardware features needed to carry out all of the designated station functions were carefully itemized. A thorough analysis of the current system's software routines served as a model for the length of the counterpart routines needed to perform similar primitive system operations in the projected system. The total length of software for a given station gave the following information: 1) the software writing cost estimate, evaluated on the basis of 1 effective line of final code per hour of programmer time, and 2) the hardware memory requirements for that station in order for that station to keep all its software in memory at once.

It was assumed for simplicity's sake that the length of a software routine in 8/A version was generally compatible to that of the 8080 version even though the coding is different. Also, the current DEC routines were weighted on the basis of the estimated revision work they need in order to adapt to the projected system; 0, 25, 50, 75, and 100 per cent rewriting factors were used to assess this adaptation cost. These findings are presented in the summary under "Software Development Costs".

The minimal system referred to in the summary consists of only one station of each kind, and its projected cost is the sum of the software and hardware development costs for each station, plus additional hardware. This additional hardware refers to ophthalmic equipment needed by the doctor, the optician, etc.

The optimal system is obtained by adding in the hardware costs (minus design) of 2 additional case history stations, 1 additional pathology detection station, 1 additional objective refractor, and 1 additional subjective refractor station. No extra software development costs are incurred because the routines are modular and the initial development of the central file and system development station allows for easy addition of stations.

No striking differences exist in the two systems to clearly imply one over the other, but certain recommendations do seem in order. The PDP-8/A would most likely be suitable for prototype development in view of the ease in extending current software to the optimal system. Construction costs were estimated on the basis of wire-wrap procedures, however, and should larger-scale production of the system become a desirable step, the fabrication efficiencies of small chip counts and printed circuit techniques would seem to indicate that the 8080 version - or the state-of-the-art LSI microprocessor available in the future - would be the most cost-efficient choice.

Finally, although the difference in estimated costs of te two versions is $50,000, this figure is only a relative difference of 10% between the two systems. Developments in the next year or two could easily change this relative cost balance on the hardware side of the picture, although the difference in software development costs is seen to be more fixed, leaning in favor of the PDP-8/A.

Regarding the overall system architecture, the choice of a distributed system over a large centralized computer is seen to be well-founded. Current research in computer science as well as existing applications confirm the functional efficiencies of spreading the work load of a system among levels of control and intelligence which conform to the inherent structure of the task. Furthermore, Large Scale Integration (LSI) makes such applications economically feasible and attractive. Writing in the Compcon 72 Digest of Papers: Innovative Architecture, E.D. Jensen of the Honeywell Systems and Research Center states that "LSI is facilitating a reorientation in digital system architecture by enabling the traditionally separate functions of processing...and storage to be combined. At the same time, this composite can be divided into entities dedicated to specific purposes and distributed through-out the system." Similarly, Earl Joseph, staff scientist for UNIVAC Division, Sperry Rand Corporation, comments that "this dispersion of computer functions, distributed where data is handled - processed, collected, or disseminated - now becomes practical", and adds, "no longer will we convert to a new system via the path of buying a total new computer - rather we will add new modules, without self-obsoleting the system".

Such architectures have already been proposed for specific systems in other studies. An example is the network which the Office of Medical Information Systems at the University of California Medical Center (San Francisco) designated as clearly the best choice for a hospital information processing system. "A number of semi-autonomous application modules, each utilizing its own mini-computer, are configured into a network to provide an information processing system to satisfy the needs of the hospital. The network configuration of small computers offers unique capabilities as well as operating economies unavailable in large processor-based systems".

Although the initial development cost of the projected optimal system prototype is appreciable, the cost of reproducing the system in manufacturing economies of scale and production techniques brings the anticipated cost per patient nearer the competitive break-even point vis-a-vis the traditionally staffed clinic. The continued development of computer-assisted systems in this and other areas offers great hope for extending the quality of professional services to larger numbers of people in society - especially as the power of technology increases as its cost decreases.

REFERENCES

Christy, P.R., Henley, R.R. and Blois, M.S. A Net work Structured Hospital Information System. In, CompCon 73 Digest of Papers: Computing Networks.

Jensen, E.D. Mixed-mode and Multi-dimensioned Memory. In, CompCon 72 Digest of Papers: Innovative Architecture.

Joseph, E. Future Computer Architecture — Polysystems. In, CompCon 72 Digest of Papers: Innovative Architecture.

LEGENDS

TABLE 1

EVALUATION CRITERIA AND PERFORMANCE CONSTRAINTS

I. OVERALL SYSTEM PERFORMANCE

A. Numerous real-time tests and processes must be simultaneously controlled in physically disjointed locations.

B. The information which each process furnishes must be collected and coordinated with patient files in a central file.

C. Access to the central file must be controlled (file protection).

D. The system should respond to a user request or a patient reaction in less than two seconds.

II. USER INTERFACE

A. No technical computer skills should be required for operation of the system.

B. The optometrist should be able to modify the number and/or sequence of test routines with a simple command language.

C. Some degree of error logging and user notification of errors should be incorporated into the system.

III. SYSTEM DEVELOPMENT USER INTERFACE

A. Tools for software development must be provided: compilers, assemblers, linking loaders, necessary secondary storage, etc.

B. Different configurations of test stations must be possible in order to test effects on patient flow.

IV. RELIABILITY

A. Operational faults in any one station should not interfere with any other part of the system.

B. All functional units which are central to system operation must have a back-up capability with soft-error recovery and hardware replacement modules.

V. MAINTENANCE

A. The design of system components should incorporate modularity whereever appropriate to allow for relative ease in debugging and replacement.

B. Some degree of automatic fault diagnosis is desirable.

TABLE 1 Cont.

VI. FUTURE OPTIONS

A. The design must permit adding components aimed at improving the power of performance of the system (memory, test stations, etc).

B. Some attention must be given to the feasibility of applying production techniques to the manufacture of the system, and these comsiderations should be included in the design where appropriate.

TABLE 2

CENTRALIZED-DISTRIBUTED SPECTRUM OF SYSTEM INTELLIGENCE

LEVEL	CENTRAL ROLE	REMOTE ROLE	EXAMPLES
1	COMPLETE CONTROL	COMPLETE HELPLESSNESS	
	Entire work load on central site, including mechanical control	Responds to central manipulation	Vector-refresh displays, D/A process control with position feedback, etc.
2	INFORMATION CONTROL	SOME INTELLIGENCE	
	Able to rely on remote unit to control physical parameters; needs to specify stream of information	Able to translate simple central intentions	Character-generating displays with local buffers; DEC I/O Transfer instructions to device cards
3	SUB-TASK COORDINATION	SOME INDEPENDENCE	
	Central unit instructs other units to accomplish sub-tasks in the course of a larger job	Able to interpret the sub-task request and carry out the sequence of operations needed under a variety of conditions which may arise	Block-transfer units using direct memory access such as disk controllers or I/O processors; concentrators and message processors; other special processor configurations in multi-processor, high-level language systems
4	EASY-GOING SUPERVISION	SELF DIRECTION	
	Central unit takes care of other units' external needs when they arise and prevents interference among units	Unit able to control its own activity, interacting with central unit and system resources in a cooperative alliance	Computer networks self-optimizing; load-sharing, public information and other futuristic systems; smaller scale networks of dedicated mini's and microprocessors

TABLE 3

DESIGN CONSIDERATIONS OF CENTRALIZED-DISTRIBUTED TRADE-OFFS

AREA	CENTRALIZED	DISTRIBUTED
Hardware performance	Requires powerful computer	Processing unit can be a microprocessor
Software development	Requires considerable software overhead (operating system for user-specified hardware configurations, time-sharing or multiprogramming software needed)	Requires file system network software, standard small computer software at local stations
Reliability	Entire system dependent upon one unit	Failures in different stations have no effect on other units, with the exception of the network
Maintenance	System down until problem is located and corrected	Module replacement enables quick servicing
Flexibility	Power of processing unit places a fixed ceiling on system performance; extensive software needed for reconfiguring system	Network of smaller processors facilitates tailoring size of system to performance needs

SIMPLIFIED REFRACTOR II SYSTEM
ORGANIZATION

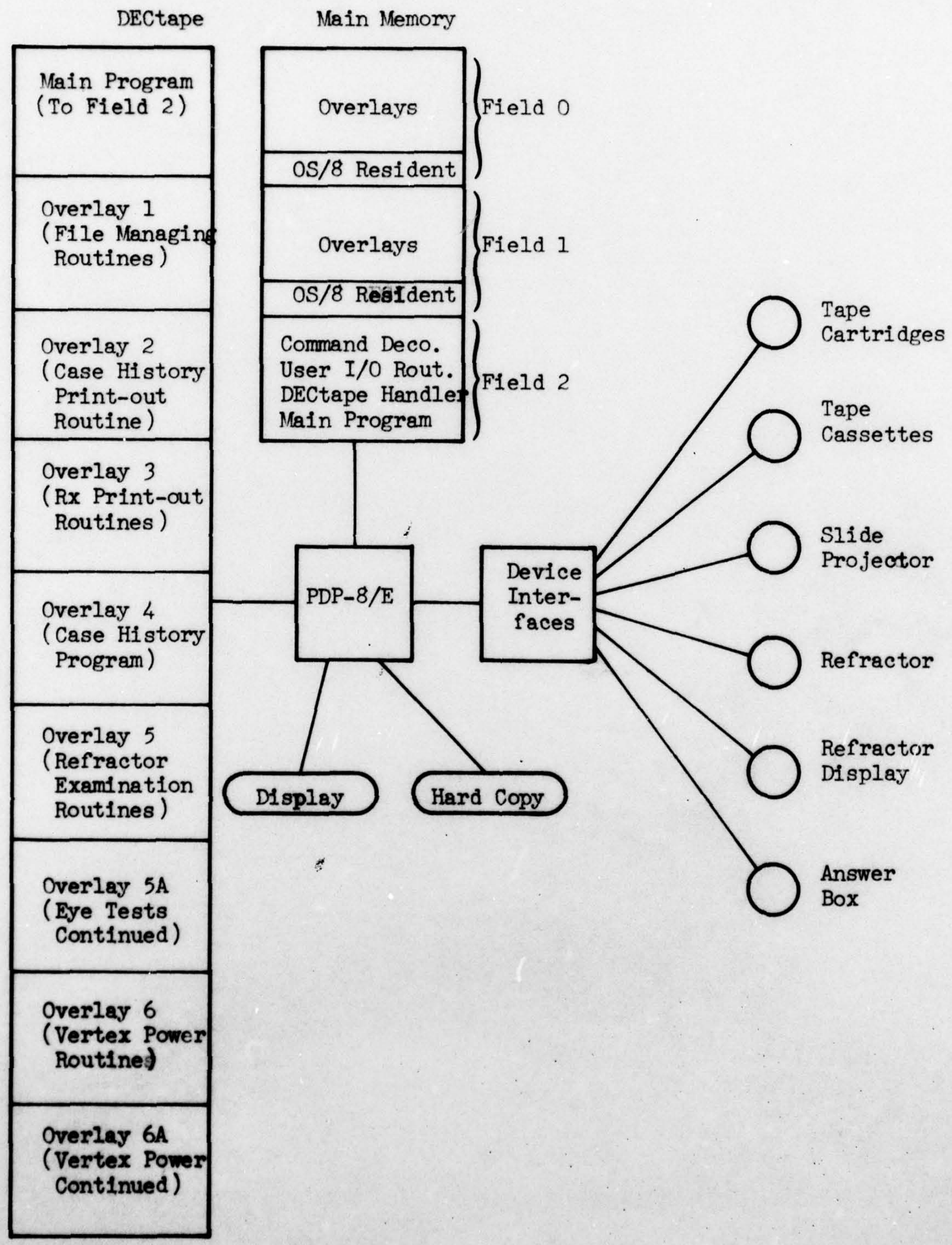

DECtape
Main Memory
Main Program (To Field 2)
Overlay 1 (File Managing Routines)
Overlay 2 (Case History Print-out Routine)
Overlay 3 (Rx Print-out Routines)
Overlay 4 (Case History Program)
Overlay 5 (Refractor Examination Routines)
Overlay 5A (Eye Tests Continued)
Overlay 6 (Vertex Power Routines)
Overlay 6A (Vertex Power Continued)
Overlays
OS/8 Resident
Field 0
Overlays
OS/8 Resident
Field 1
Command Deco.
User I/O Rout.
DECtape Handler
Main Program
Field 2
PDP-8/E
Device Inter-faces
Display
Hard Copy
Tape Cartridges
Tape Cassettes
Slide Projector
Refractor
Refractor Display
Answer Box

Fig.
1

CENTRALIZED IMPLEMENTATION

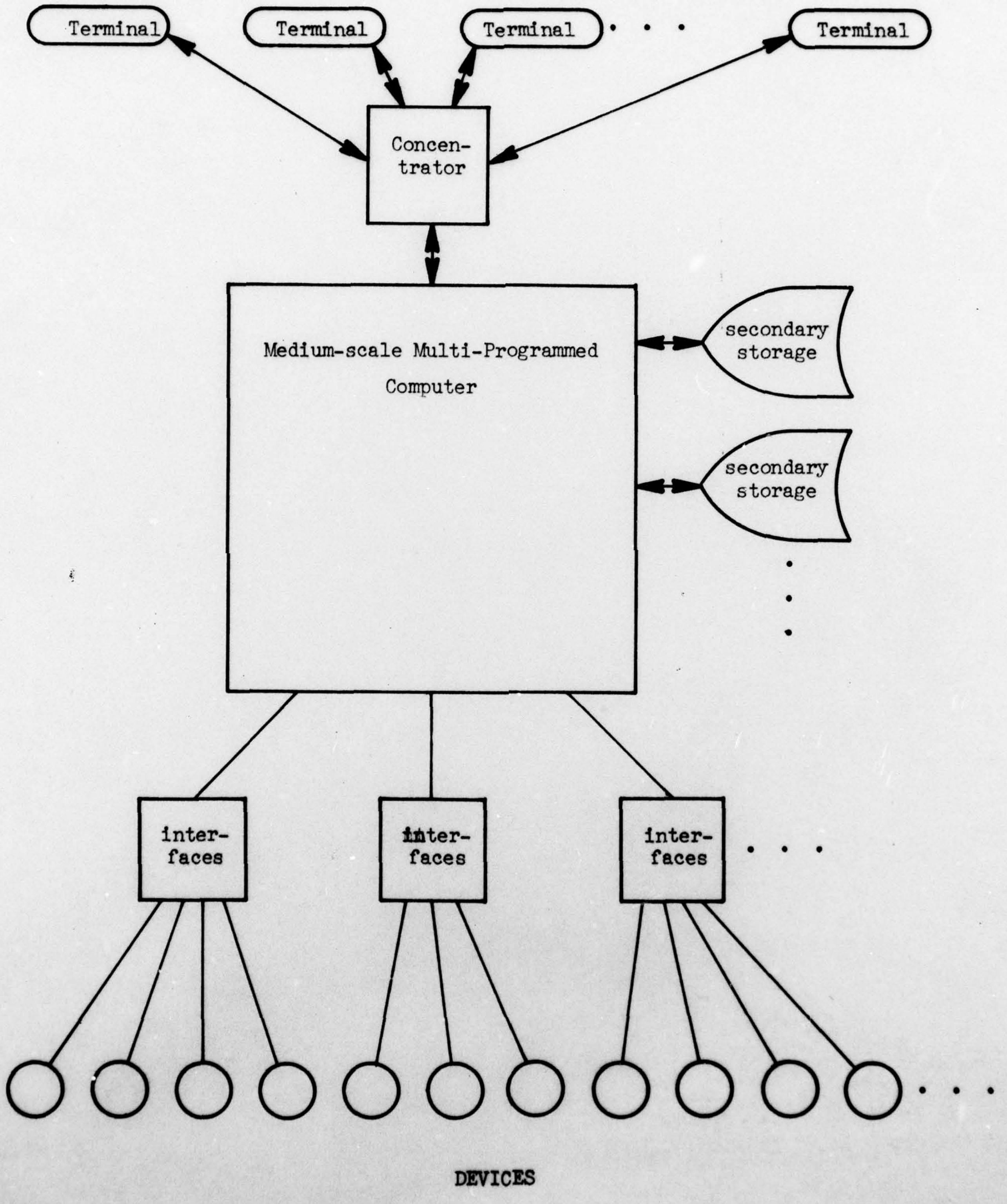


Fig.
2

DISTRIBUTED IMPLEMENTATION

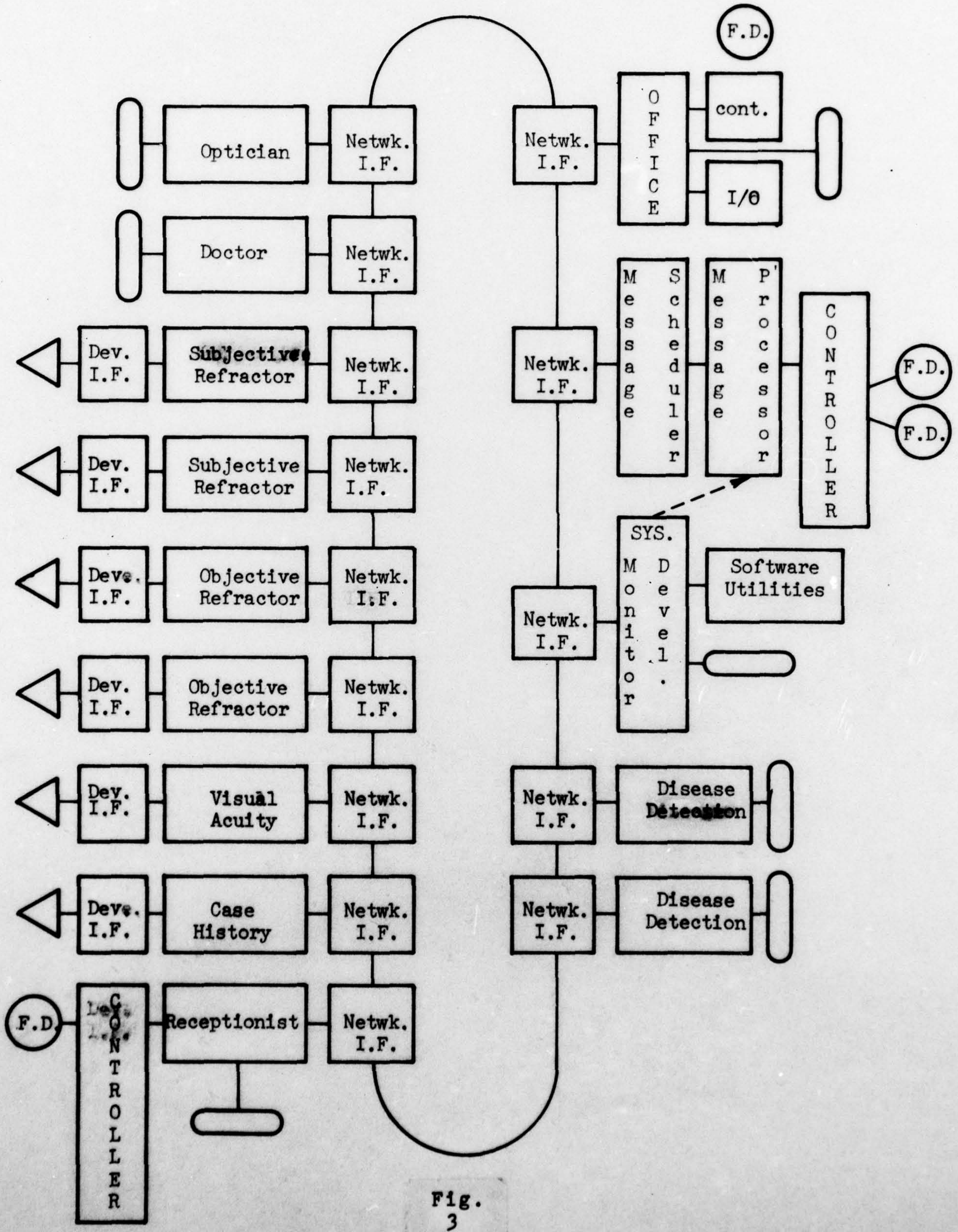


Fig. 3

HARDWARE/SOFTWARE RELATIONSHIPS IN A
GENERALIZED PROCESSOR STATION

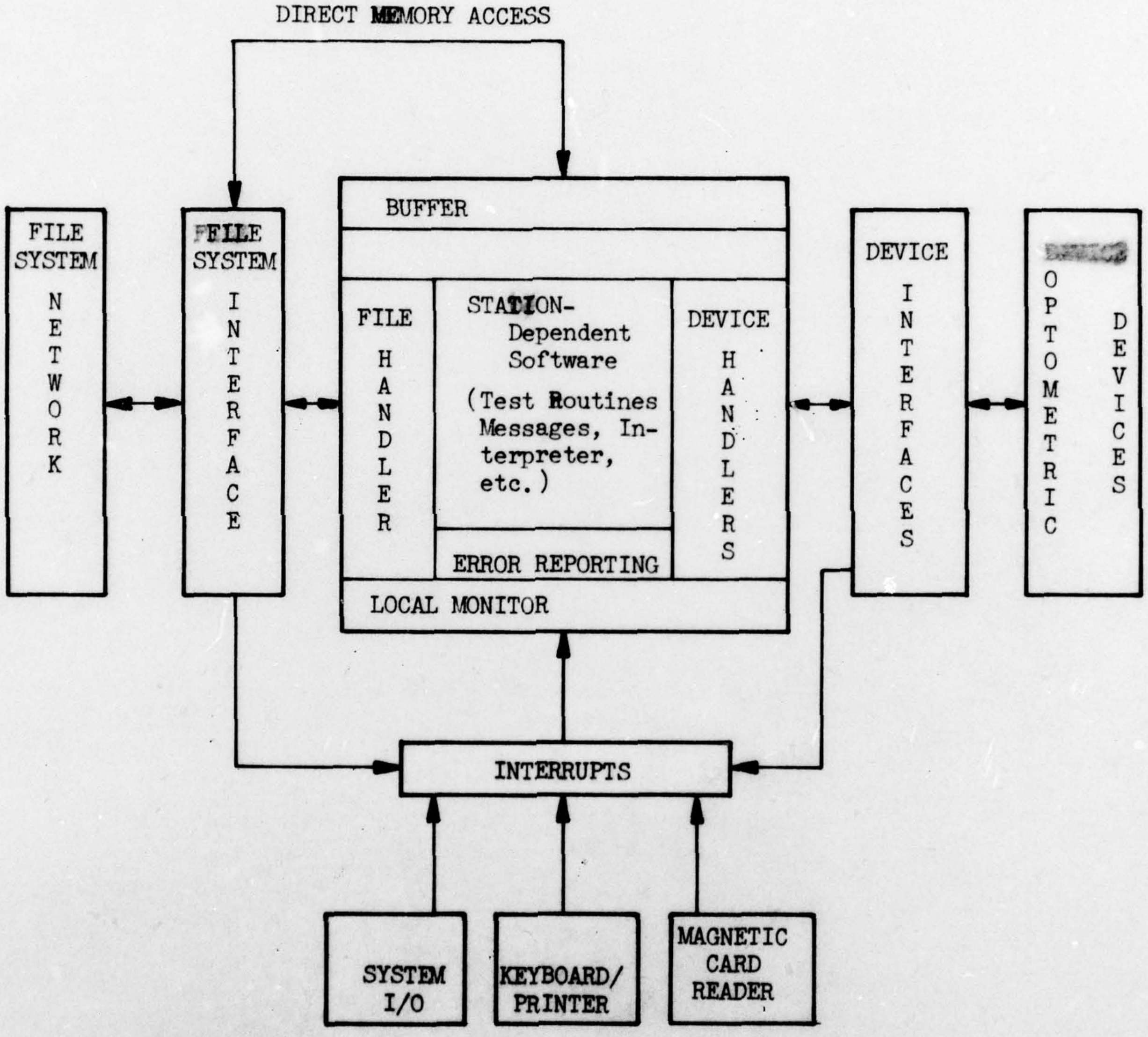


Fig.
4

DISTRIBUTION LIST

4 copies	HQDA (SGRD-AJ) Fort Detrick Frederick, MD. 21701
12 copies	Defense Documentation Center (DDC) ATTN: DDC-DCA Cameron Station Alexandria, Virginia 22314
1 copy	Dean School of Medicine Uniformed Services University of the Health Sciences 4301 Jones Bridge Road Bethesda, Maryland 20014
1 copy	Superintendent Academy of Health Sciences, US Army ATTN: AHS-COM Fort Sam Houston, Texas 78234